

Derandomization for k -submodular maximization

Hiroki Oshima

Department of Mathematical Informatics,
Graduate School of Information Science and Technology,
The University of Tokyo

October 26, 2016

Abstract

Submodularity is one of the most important property of combinatorial optimization, and k -submodularity is a generalization of submodularity. Maximization of k -submodular function is NP-hard, and approximation algorithm is studied. For monotone k -submodular function, [Iwata, Tanigawa, and Yoshida 2016] gave $k/(2k-1)$ -approximation algorithm. In this paper, we give a deterministic algorithm by derandomizing that algorithm. Derandomization scheme is from [Buchbinder and Feldman 2016]. Our algorithm is $k/(2k-1)$ -approximation and polynomial-time algorithm.

1 Introduction

In this paper, we give a deterministic approximate algorithm for monotone k -submodular maximization.

DEFINITION 1.1. Let $(k+1)^V := \{(X_1, \dots, X_k) \mid X_i \subseteq V \ (i = 1, \dots, k), X_i \cap X_j = \emptyset \ (i \neq j)\}$. A function $f : (k+1)^V \rightarrow \mathbb{R}$ is called k -submodular if we have

$$f(\mathbf{x}) + f(\mathbf{y}) \geq f(\mathbf{x} \sqcap \mathbf{y}) + f(\mathbf{x} \sqcup \mathbf{y})$$

for any $\mathbf{x} = (X_1, \dots, X_k)$, $\mathbf{y} = (Y_1, \dots, Y_k) \in (k+1)^V$. Note that

$$\begin{aligned} \mathbf{x} \sqcap \mathbf{y} &= (X_1 \cap Y_1, \dots, X_k \cap Y_k) \text{ and} \\ \mathbf{x} \sqcup \mathbf{y} &= (X_1 \cup Y_1 \setminus (\bigcup_{i \neq 1} X_i \cup Y_i), \dots, X_k \cup Y_k \setminus (\bigcup_{i \neq k} X_i \cup Y_i)). \end{aligned}$$

We consider the maximization problem for k -submodular functions. It is NP-hard, then finding strict maximizer of k -submodular function is difficult in value oracle model. Hence, approximation algorithm have been studied. Input of the problem is a nonnegative k -submodular function. Note that, for any k -submodular f and any $c \in \mathbb{R}$, a function $f'(\mathbf{x}) := f(\mathbf{x}) + c$ is k -submodular. Output of the problem is $\mathbf{x} = (X_1, \dots, X_k) \in (k+1)^V$. Let input k -submodular function be f , a maximizer of f be \mathbf{o} , and output of algorithm be \mathbf{s} . Then we define approximation ratio as $f(\mathbf{s})/f(\mathbf{o})$ for deterministic algorithm, and $\mathbb{E}[f(\mathbf{s})]/f(\mathbf{o})$ for randomized algorithm.

2 Preliminary

Define a partial order \preceq on $(k+1)^V$ for $\mathbf{x} = (X_1, \dots, X_k)$ and $\mathbf{y} = (Y_1, \dots, Y_k)$ as follows:

$$\mathbf{x} \preceq \mathbf{y} \stackrel{\text{def}}{\iff} X_i \subseteq Y_i (i = 1, \dots, k).$$

Also, for $\mathbf{x} = (X_1, \dots, X_k) \in (k+1)^V$, $e \notin \bigcup_{l=1}^k X_l$, and $i \in \{1, \dots, k\}$, define

$$\Delta_{e,i}f(\mathbf{x}) = f(X_1, \dots, X_{i-1}, X_i \cup \{e\}, X_{i+1}, \dots, X_k) - f(X_1, \dots, X_k).$$

A monotone k -submodular function is k -submodular and satisfies $f(\mathbf{x}) \leq f(\mathbf{y})$ for any $\mathbf{x} = (X_1, \dots, X_k)$ and $\mathbf{y} = (Y_1, \dots, Y_k)$ in $(k+1)^V$ with $\mathbf{x} \preceq \mathbf{y}$.

Property of k -submodularity can be written as another form.

THEOREM 2.1. ([4] THEOREM 7) *A function $f : (k+1)^V \rightarrow \mathbb{R}$ is k -submodular if and only if f is orthant submodular and pairwise monotone.*

Note that orthant submodularity is to satisfy

$$\Delta_{e,i}f(\mathbf{x}) \geq \Delta_{e,i}f(\mathbf{y}) \quad (\mathbf{x}, \mathbf{y} \in (k+1)^V, \mathbf{x} \preceq \mathbf{y}, e \notin \bigcup_{l=1}^k Y_l, i \in \{1, \dots, k\}),$$

and pairwise monotonicity is to satisfy

$$\Delta_{e,i}f(\mathbf{x}) + \Delta_{e,j}f(\mathbf{x}) \geq 0 \quad (\mathbf{x} \in (k+1)^V, e \notin \bigcup_{l=1}^k X_l, i, j \in \{1, \dots, k\} (i \neq j)).$$

3 Existing randomized algorithm

To analyze k -submodular function, it is often convenient to identify $(k+1)^V$ as $\{0, 1, \dots, k\}^V$. A $|V|$ -dimensional vector $\mathbf{x} \in \{0, 1, \dots, k\}^V$ is associated with $(X_1, \dots, X_n) \in (k+1)^V$ by $X_i = \{e \in V \mid \mathbf{x}(e) = i\}$.

3.1 Algorithm framework

In this section, we see the framework to maximize k -submodular functions (Algorithm 1 [3]). [2] and [4] used it with specific distribution.

Algorithm 1 ([3] Algorithm 1)

Input: A nonnegative k -submodular function $f : \{0, 1, \dots, k\}^V \rightarrow \mathbb{R}_+$.

Output: A vector $\mathbf{s} \in \{0, 1, \dots, k\}^V$.

$\mathbf{s} \leftarrow \mathbf{0}$.

Denote the elements of V by $e^{(1)}, \dots, e^{(n)}$ ($|V| = n$).

for $j = 1, \dots, n$ **do**

 Set a probability distribution $p^{(j)}$ over $\{1, \dots, k\}$.

 Let $\mathbf{s}(e^{(j)}) \in \{1, \dots, k\}$ be chosen randomly with $\Pr[\mathbf{s}(e^{(j)}) = i] = p_i^{(j)}$.

end for

return \mathbf{s}

Algorithm 1 is not only used for monotone functions. However, in this paper, we only use it for monotone functions.

Now we define some variables to see Algorithm 1. Let \mathbf{o} be an optimal solution, and we write $\mathbf{s}^{(j)}$ as \mathbf{s} at j -th iteration. Let other variables be as follows:

$$\begin{aligned} \mathbf{o}^{(j)} &= (\mathbf{o} \sqcup \mathbf{s}^{(j)}) \sqcup \mathbf{s}^{(j)} \quad , \quad \mathbf{t}^{(j-1)}(e) = \begin{cases} \mathbf{o}^{(j)}(e) & (e \neq e^{(j)}) \\ 0 & (e = e^{(j)}) \end{cases} \\ y_i^{(j)} &= \Delta_{e^{(j)}, i} f(\mathbf{s}^{(j-1)}) \quad , \quad a_i^{(j)} = \Delta_{e^{(j)}, i} f(\mathbf{t}^{(j-1)}) \end{aligned}$$

Algorithm 1 satisfies following lemma.

LEMMA 3.1. ([3] LEMMA 2.1.)

Let $c \in \mathbb{R}_+$. Conditioning on $\mathbf{s}^{(j-1)}$, suppose that

$$\sum_{i=1}^k (a_{i^*}^{(j)} - a_i^{(j)}) p_i^{(j)} \leq c \sum_{i=1}^k (y_i^{(j)} p_i^{(j)})$$

holds for each j with $1 \leq j \leq n$, where $i^* = \mathbf{o}(e^{(j)})$. Then $\mathbb{E}[f(\mathbf{s})] \geq \frac{1}{1+c} f(\mathbf{o})$.

3.2 A randomized algorithm for monotone functions

In [3], a randomized $\frac{k}{2k-1}$ -approximation algorithm for monotone k -submodular function (Algorithm 2) is shown.

Algorithm 2 ([3] Algorithm 3)

Input: A monotone k -submodular function $f : \{0, 1, \dots, k\}^V \rightarrow \mathbb{R}_+$.

Output: A vector $\mathbf{s} \in \{0, 1, \dots, k\}^V$.

$\mathbf{s} \leftarrow \mathbf{0}, t \leftarrow k - 1$.

Denote the elements of V by $e^{(1)}, \dots, e^{(n)}$ ($|V| = n$).

for $j = 1, \dots, n$ **do**

$y_i^{(j)} \leftarrow \Delta_{e^{(j)}, i} f(\mathbf{s}) \quad (1 \leq i \leq k)$.

$\beta \leftarrow \sum_{i=1}^k (y_i^{(j)})^t$.

if $\beta \neq 0$ **then** $p_i^{(j)} \leftarrow (y_i^{(j)})^t / \beta \quad (1 \leq i \leq k)$.

else

$p_1^{(j)} = 1, p_i^{(j)} = 0 \quad (i = 2, \dots, k)$.

end if

 Let $\mathbf{s}(e^{(j)}) \in \{1, \dots, k\}$ be chosen randomly with $\Pr[\mathbf{s}(e^{(j)}) = i] = p_i^{(j)}$.

end for

return \mathbf{s}

Algorithm 2 runs in polynomial time. The approximation ratio of Algorithm 2 satisfies the theorem below.

THEOREM 3.1. ([3] THEOREM 2.2.) *Let \mathbf{o} be a maximizer of a monotone k -submodular function f and let \mathbf{s} be the output of Algorithm 2. Then $\mathbb{E}[f(\mathbf{s})] \geq \frac{k}{2k-1} f(\mathbf{o})$.*

In the proof of this theorem (see [3]), the inequality of Lemma 3.1 is proved with $c = 1 - \frac{1}{k}$. We get $a_i \geq 0$ ($\forall i \in \{1, \dots, k\}$) from monotonicity, and $a_i \leq y_i$ ($\forall i \in \{1, \dots, k\}$) from orthant submodularity. Hence, the inequality

$$\sum_{i \neq i^*} (y_{i^*}^{(j)} p_i^{(j)}) \leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^k (y_i^{(j)} p_i^{(j)}) \quad (1)$$

is used. The inequality of Lemma 3.1 is satisfied when the inequality (1) is valid.

4 Deterministic algorithm

In this section, we show a polynomial-time deterministic algorithm for maximizing monotone k -submodular functions. Our algorithm is Algorithm 3. Algorithm 3 is a derandomized version of Algorithm 2. We note derandomization scheme of this algorithm is from [1].

In the algorithm, we construct a distribution \mathcal{D} satisfies $\mathbb{E}_{\mathbf{s} \sim \mathcal{D}}[f(\mathbf{s})] \geq \frac{k}{2k-1} f(\mathbf{o})$. Then algorithm outputs the best solution in $\text{supp}(\mathcal{D}) := \{\mathbf{s} \mid$

Algorithm 3 Deterministic Algorithm

Input: A monotone k -submodular function $f : \{0, 1, \dots, k\}^V \rightarrow \mathbb{R}_+$.

Output: A vector $\mathbf{s} \in \{0, 1, \dots, k\}^V$.

$\mathcal{D}_0 \leftarrow (1, \mathbf{0})$, ($\mathcal{D} = \{(p, \mathbf{s}) \mid \mathbf{s} \in (k+1)^V, 0 \leq p \leq 1\}$ ($\sum_{\mathbf{s} \in \mathcal{D}} p = 1$)).

Denote the elements of V by $e^{(1)}, \dots, e^{(n)}$ ($|V| = n$).

for $j = 1, \dots, n$ **do**

$y_i(\mathbf{s}) \leftarrow \Delta_{e^{(j)}, i} f(\mathbf{s})$ ($\forall \mathbf{s} \in \text{supp}(\mathcal{D}_{j-1}), i \in \{1, \dots, k\}$).

Find an extreme point solution $(p_{i,\mathbf{s}})_{i=1,\dots,k, \mathbf{s} \in \text{supp}(\mathcal{D}_{j-1})}$ of the following linear formulation:

$$\left(1 - \frac{1}{k}\right) \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} \left[\sum_{i=1}^k p_{i,\mathbf{s}} y_i(\mathbf{s}) \right] \geq \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} [(1 - p_{l,\mathbf{s}}) y_l(\mathbf{s})] \quad (2)$$

$$(l \in \{1, \dots, k\})$$

$$\sum_{i=1}^k p_{i,\mathbf{s}} = 1 \quad (\forall \mathbf{s} \in \text{supp}(\mathcal{D}_{j-1})) \quad (3)$$

$$p_{i,\mathbf{s}} \geq 0 \quad (\forall \mathbf{s} \in \text{supp}(\mathcal{D}_{j-1}), i \in \{1, \dots, k\}). \quad (4)$$

Construct a new distribution \mathcal{D}_j :

$$\mathcal{D}_j \leftarrow \bigcup_{i=1}^k \{(p_{i,\mathbf{s}} \cdot \Pr_{\mathcal{D}_{j-1}}[\mathbf{s}], \mathbf{s}_{e^{(j)}, i}) \mid \mathbf{s} \in \text{supp}(\mathcal{D}_{j-1}), p_{i,\mathbf{s}} > 0\} \quad (5)$$

$$\left(\mathbf{s}_{e^{(j)}, i}(e) = \begin{cases} \mathbf{s}(e) & (e \neq e^{(j)}) \\ i & (e = e^{(j)}) \end{cases} \right).$$

end for

return $\arg \max_{\mathbf{s} \in \text{supp}(\mathcal{D}_n)} \{f(\mathbf{s})\}$

$(p, \mathbf{s}) \in \mathcal{D}\}$. We can see the right hand side of (2) is the expected value of the left hand side of (1) for $\mathbf{s} \sim \mathcal{D}_{j-1}$. Also the left hand side of (2) is the expected value of the right hand side of (1) with $c = 1 - 1/k$. From (3) and (4), \mathcal{D}_j in (5) is constructed as a distribution.

Algorithm 3 satisfies the same approximation ratio as Algorithm 2.

THEOREM 4.1. *Let \mathbf{o} be a maximizer of a monotone nonnegative k -submodular function f and let \mathbf{z} be the output of Algorithm 3. Then $f(\mathbf{z}) \geq \frac{k}{2k-1} f(\mathbf{o})$.*

Proof. We consider the j -th iteration. From (5), we get

$$\begin{aligned} \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} \left[\sum_{i=1}^k p_{i,\mathbf{s}} y_i(\mathbf{s}) \right] &= \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} \left[\sum_{i=1}^k p_{i,\mathbf{s}} (f(\mathbf{s}_{e^{(j)},i}) - f(\mathbf{s})) \right] \\ &= \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} \left[\sum_{i=1}^k p_{i,\mathbf{s}} f(\mathbf{s}_{e^{(j)},i}) - f(\mathbf{s}) \right] \\ &= \mathbb{E}_{\mathbf{s}' \sim \mathcal{D}_j} [f(\mathbf{s}')] - \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} [f(\mathbf{s})]. \end{aligned} \quad (6)$$

Now, we consider $\mathbf{o}[\mathbf{s}] := (\mathbf{o} \sqcup \mathbf{s}) \sqcup \mathbf{s}$. Define the variables as follows:

$$\begin{aligned} \mathbf{t}[\mathbf{s}](e) &= \begin{cases} \mathbf{o}[\mathbf{s}](e) & (e \neq e^{(j)}) \\ 0 & (e = e^{(j)}) \end{cases} \\ a_i(\mathbf{s}) &= \Delta_{e^{(j)},i} f(\mathbf{t}[\mathbf{s}]) \end{aligned}$$

Then we have

$$f(\mathbf{o}[\mathbf{s}]) - f(\mathbf{o}[\mathbf{s}_{e^{(j)},i}]) = a_{i^*}(\mathbf{s}) - a_i(\mathbf{s}) \quad (i^* = \mathbf{o}(e^{(j)})) \quad (7)$$

From monotonicity and orthant submodularity of f , we have

$$a_{i^*}(\mathbf{s}) - a_i(\mathbf{s}) \leq y_{i^*}(\mathbf{s}). \quad (8)$$

From (7) and (8), we get

$$\begin{aligned} \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} [f(\mathbf{o}[\mathbf{s}])] - \mathbb{E}_{\mathbf{s}' \sim \mathcal{D}_j} [f(\mathbf{o}[\mathbf{s}'])] &= \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} \left[\sum_{i=1}^k p_{i,\mathbf{s}} f(\mathbf{o}[\mathbf{s}]) - f(\mathbf{o}[\mathbf{s}_{e^{(j)},i}]) \right] \\ &= \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} \left[\sum_{i=1}^k p_{i,\mathbf{s}} (f(\mathbf{o}[\mathbf{s}]) - f(\mathbf{o}[\mathbf{s}_{e^{(j)},i}])) \right] \\ &= \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} \left[\sum_{i \neq i^*} p_{i,\mathbf{s}} (a_{i^*}(\mathbf{s}) - a_i(\mathbf{s})) \right] \\ &\leq \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} \left[\sum_{i \neq i^*} p_{i,\mathbf{s}} (y_{i^*}(\mathbf{s})) \right] \\ &= \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} [(1 - p_{i^*,\mathbf{s}}) (y_{i^*}(\mathbf{s}))]. \end{aligned} \quad (9)$$

$p_{i,\mathbf{s}}$ satisfies (2) for all $l \in 1, 2, \dots, k$. Hence we get

$$\left(1 - \frac{1}{k}\right) (\mathbb{E}_{\mathbf{s}' \sim \mathcal{D}_j} [f(\mathbf{s}')] - \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} [f(\mathbf{s})]) \geq \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_{j-1}} [f(\mathbf{o}[\mathbf{s}])] - \mathbb{E}_{\mathbf{s}' \sim \mathcal{D}_j} [f(\mathbf{o}[\mathbf{s}'])] \quad (10)$$

from (6) and (9). By the summation of (10), we get

$$\left(1 - \frac{1}{k}\right) (\mathbb{E}_{\mathbf{s}' \sim \mathcal{D}_n} [f(\mathbf{s}')] - \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_0} [f(\mathbf{s})]) \geq \mathbb{E}_{\mathbf{s} \sim \mathcal{D}_0} [f(\mathbf{o}[\mathbf{s}])] - \mathbb{E}_{\mathbf{s}' \sim \mathcal{D}_n} [f(\mathbf{o}[\mathbf{s}'])] . \quad (11)$$

Note that $\mathbf{o}[\mathbf{s}'] = \mathbf{s}'$ for $\mathbf{s}' \in \text{supp}(\mathcal{D}_n)$, and $\mathbf{o}[\mathbf{s}] = \mathbf{o}$ for $\mathbf{s} \in \text{supp}(\mathcal{D}_0)$. Now we have

$$\begin{aligned} f(\mathbf{o}) &\leq \left(2 - \frac{1}{k}\right) \mathbb{E}_{\mathbf{s}' \sim \mathcal{D}_n} [f(\mathbf{s}')] - \left(1 - \frac{1}{k}\right) f(\mathbf{o}) \\ &\leq \left(2 - \frac{1}{k}\right) \mathbb{E}_{\mathbf{s}' \sim \mathcal{D}_n} [f(\mathbf{s}')] \\ &\leq \left(2 - \frac{1}{k}\right) \max_{\mathbf{s}' \in \text{supp}(\mathcal{D}_n)} \{f(\mathbf{s}')\} \end{aligned}$$

□

The algorithm performs polynomial number of value oracle queries.

THEOREM 4.2. *Algorithm 3 return a solution after $O(n^2 k^2)$ value oracle queries.*

Proof. Algorithm uses oracle to calculate $y_i(\mathbf{s})$. At j -th iteration, the number of $y_i(\mathbf{s})$ is $k|\mathcal{D}_{j-1}|$. From (5), $|\mathcal{D}_j|$ equals the number of $p_{i,\mathbf{s}} \neq 0$. Then we have to consider $p_{i,\mathbf{s}} \neq 0$ at j -th iteration.

By the definition, $(p_{i,\mathbf{s}})_{i=1,\dots,k, \mathbf{s} \in \text{supp}(\mathcal{D}_{j-1})}$ is an extreme point solution of (2), (3), and (4). Note that, we can get a solution by setting $(p_{i,\mathbf{s}})$ as the solution of [3] for all $\mathbf{s} \in \text{supp}(\mathcal{D}_{j-1})$. We can also see the feasible region of (2), (3), and (4) is bounded. Then the extreme point solution exists.

Let $|\mathcal{D}_{j-1}| = m$. By $(p_{i,\mathbf{s}})_{i=1,\dots,k, \mathbf{s} \in \text{supp}(\mathcal{D}_{j-1})} \in \mathbb{R}^{km}$ and k equalities of (3), $km - k$ inequalities are tight at any extreme point solution. (2) have m inequalities and (4) have km inequalities. Then, at least $km - k - m$ inequalities of (4) are tight. Hence, the number of $p_{i,\mathbf{s}} \neq 0$ is at most $m + k$.

Now we have $|\mathcal{D}_j| \leq |\mathcal{D}_{j-1}| + k$. We can also see $|\mathcal{D}_j| \leq jk + 1$. Then the number of value oracle queries is

$$\sum_{j=1}^n k|\mathcal{D}_{j-1}| \leq \sum_{j=1}^n k(jk + 1)$$

□

We use LP for searching an extreme point solution, and it runs in polynomial-time. Hence Algorithm 3 is a polynomial-time algorithm.

5 Conclusion

We showed a derandomized algorithm for monotone k -submodular maximization. It is $\frac{k}{2k-1}$ -approximation and polynomial-time algorithm.

References

- [1] N. Buchbinder and M. Feldman. Deterministic algorithms for submodular maximization problems. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 392–403. SIAM, 2016.
- [2] S. Iwata, S. Tanigawa, and Y. Yoshida. Bisubmodular function maximization and extensions. Technical report, Technical Report METR 2013-16, The University of Tokyo, 2013.
- [3] S. Iwata, S. Tanigawa, and Y. Yoshida. Improved approximation algorithms for k -submodular function maximization. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 404–413. SIAM, 2016.
- [4] J. Ward and S. Živný. Maximizing k -submodular functions and beyond. *ACM Trans. Algorithms*, 12(4):47:1–47:26, August 2016.